Comparative Analysis of Kernel Methods for Statistical Shape Learning

Yogesh Rathi, Samuel Dambreville, and Allen Tannenbaum *

Georgia Institute of Technology, Atlanta, GA, 30332, USA
{yogesh.rathi},{samuel.dambreville},{tannenba}@bme.gatech.edu

Abstract. Prior knowledge about shape may be quite important for image segmentation. In particular, a number of different methods have been proposed to compute the statistics on a set of training shapes, which are then used for a given image segmentation task to provide the shape prior. In this work, we perform a comparative analysis of shape learning techniques such as linear PCA, kernel PCA, locally linear embedding and propose a new method, kernelized locally linear embedding for doing shape analysis. The surfaces are represented as the zero level set of a signed distance function and shape learning is performed on the embeddings of these shapes. We carry out some experiments to see how well each of these methods can represent a shape, given the training set.

1 Introduction

Image Segmentation has been a topic of extensive research in the computer vision community [1–4].One of the challenges in the field of image segmentation is the incorporation of prior shape knowledge in the segmentation process [5]. Many different methods (using both parameterized or implicit representation of shapes) have been proposed [6–10] to perform statistical shape analysis on a given set of training shapes. In this work, we perform a comparative analysis of several key techniques such as linear PCA (LPCA), kernel PCA (KPCA), locally linear embedding (LLE), and then propose a new method, kernelized locally linear embedding (KLLE) which will be compared with the aforementioned techniques.

There is a large body of literature available for representing a curve or surface using parameterized as well as implicit methods; see [3, 11, 12] and the references therein. A number of methods have been proposed, using these representations, to study the statistical variations in a given set of training shapes. Cootes *et al.* [6] developed a parametric point distribution model for describing the segmenting curve by using linear combinations of eigenvectors that reflect variations from

* This research was supported by grants from NSF, NIH (NAC P41 RR-13218 through Brigham and Women's Hospital), AFOSR, ARO, MURI, and MRI-HEL, and the Technion, Israel Insitute of Technology. This work was done under the auspices of the National Alliance for Medical Image Computing (NAMIC), funded by the National Institutes of Health through the NIH Roadmap for Medical Research, Grant U54 EB005149.

the mean shape. In [13], Wang and Staib developed a statistical point model by applying linear PCA to the covariance matrices that capture the statistical variations of the landmark points. Recently, Leventon [9] proposed a more general model wherein PCA was performed on a set of signed distance functions. Kernel PCA has been successfully used by the machine learning community for pattern recognition and image denoising [14]. A Gaussian kernel was used by Cremers *et al.* [8] for learning shape statistics in the kernel space to provide shape prior for segmentation tasks. Finding the pre-image of the projection in the kernel space is one of the challenging tasks in visualizing and computing the performance of these kernel based techniques. In this work, we use the method proposed by [15] to find the pre-image of the projection in the KPCA space and compare it with LPCA.

Locally linear embedding has been widely used for dimensionality reduction and extracting out the nonlinearities in the training data set. In this work, we use LLE to represent a given shape using a linear combination of its nearest neighbors. We further develop this algorithm and propose a new method to perform LLE in the kernel space, called KLLE, and show that it is better than LLE, LPCA and is comparable to KPCA in terms of performance but with fewer computations. Of course, the literature reviewed above is by no means exhaustive. We merely want to point out a new technique that has some attractive features which may act as an alternative to some already existing methodologies.

The rest of the paper is organized as follows: In the next section we briefly describe LPCA, KPCA, LLE and provide details about KLLE. In section 3, we show with examples how well each of these methods perform on a given data set. Section 4 gives conclusion and future research directions.

2 Statistical Models

This section briefly describes each of the shape learning techniques used later in the sequel. Let τ be the training set $\tau = \{\phi_1, \phi_2, \ldots, \phi_n\}$ consisting of n signed distance functions (SDF) with the shapes represented by the corresponding zero level sets. It is assumed that all the ϕ_i's are aligned using a suitable method of registration [6].

2.1 Linear PCA

Linear PCA is widely used to learn the statistical variations of a given set of data (shapes, in our case). LPCA assumes that the set of permissible shapes form a Gaussian distribution, i.e., all possible shapes can be written as a linear combination of a set of eigenshapes obtained by doing principal component analysis on the training data set [10, 9]. The eigenshapes can be obtained as follows: Let ϕ_i represent the signed distance function corresponding to the surface S_i. The mean surface, μ, is computed by taking the mean of the signed distance functions, $\mu = \frac{1}{n}\sum \phi_i$. The variance in shape is computed using PCA, i.e., the mean shape μ is subtracted from each ϕ_i to create a mean-offset map $\bar{\phi}_i$. Each

such map, $\bar{\phi}_i$, is placed as a column vector in an $N^d \times n$-dimensional matrix M, where $\phi_i \in \mathbf{R}^{N^d}$. Using Singular Value Decomposition (SVD), the covariance matrix $\frac{1}{n}MM^T$ is decomposed as:

$$U\Sigma U^T = \frac{1}{n}MM^T \tag{1}$$

where U is a matrix whose column vectors represent the set of orthogonal modes of shape variation (eigenshapes) and Σ is a diagonal matrix of corresponding eigenvalues. An estimate of a novel shape Φ of the same class of object can be obtained from m principal components using an m-dimensional vector of coefficients,

$$\alpha = U_m^T(\Phi - \mu), \tag{2}$$

where U_m is a matrix consisting of the first m columns of U. Given the coefficients α, an estimate of the shape Φ, namely $\tilde{\Phi}$, can be obtained as [9, 10]:

$$\tilde{\Phi} = U_m\alpha + \mu. \tag{3}$$

2.2 Kernel PCA

Kernel methods, in particular, kernel PCA has been the focus of research in the pattern recognition community[16, 17]. The basic idea behind these methods is to map the data in the input space $\phi \in \chi$ to a feature space F via some nonlinear map Ψ, and then apply a linear method in F to do further analysis. Kernel PCA [14] is a nonlinear feature extractor, where PCA is performed in the feature space F which is equivalent to doing nonlinear PCA in the input space χ. Since the nonlinear map Ψ is not known, a challenging problem is to find the pre-image of the projection obtained by doing PCA in the feature space F. As demonstrated by Mika [16], the exact pre-image typically may not exist and one can only settle for an approximate solution. But even this may be non-trivial as the dimensionality of the feature space can be infinite. For certain invertible kernels, this nonlinear problem can be solved using a fixed-point iteration method as proposed by Scholkopf and Mika [14, 16]. However, this method is dependent on the initial starting point and is highly susceptible to local minima. To circumvent this problem, [17] and more recently [15] proposed an algorithm to reconstruct an approximate pre-image of the projection as described briefly in the remainder of this section.

Kernel PCA performs the traditional linear PCA in the feature space corresponding to the kernel $k(.,.)$. The kernel defines the inner product between two points in the feature space, i.e., $k(\phi_1, \phi_2) =< \Psi(\phi_1), \Psi(\phi_2) >$. This fact can be used to obtain the eigenvectors in the feature space F even though the non-linear map Ψ is unknown. Analogous to linear PCA, it involves the following eigen-decomposition

$$HKH = U\Sigma U^T,$$

where, K is the kernel matrix with entries $K_{ij} = k(\phi_i, \phi_j)$, H is the centering matrix given by

$$H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T,$$

I is the $n \times n$ identity matrix, $\mathbf{1} = [11...1]^T$ is an $n \times 1$ vector, $U = [\mathbf{a}_1, ..., \mathbf{a}_n]$ with $\mathbf{a}_i = [a_{i1}, ..., a_{in}]^T$ is the matrix containing the eigenvectors and $\Sigma = diag(\lambda_1, ..., \lambda_n)$ contains the corresponding eigenvalues. Denote the mean of the Ψ-mapped data by $\bar{\Psi} = \frac{1}{n}\sum_{i=1}^{n} \Psi(\phi_i)$ and define the "centered" map $\tilde{\Psi}$ as :

$$\tilde{\Psi}(\phi) = \Psi(\phi) - \bar{\Psi}.$$

The k-th orthonormal eigenvector of the covariance matrix in the feature space can then be shown to be [14]

$$V_k = \sum_{i=1}^{n} \frac{a_{ki}}{\sqrt{\lambda_k}} \tilde{\Psi}(\phi_i).$$

Denote the projection of the Ψ-image of a test point Φ onto the k-th component by β_k. Then,

$$\beta_k = \frac{1}{\sqrt{\lambda_k}} \sum_{i=1}^{n} a_{ki} \tilde{k}(\Phi, \phi_i), \tag{4}$$

where,

$$\begin{aligned} &\tilde{k}(x, y) =< \tilde{\Psi}(x), \tilde{\Psi}(y) >= k(x, y) - \frac{1}{n}\mathbf{1}^T k_x - \frac{1}{n}\mathbf{1}^T k_y + \frac{1}{n^2}\mathbf{1}^T K \mathbf{1} \\ &\text{with} \quad k_x = [k(x, \phi_1), ..., k(x, \phi_n)]^T \end{aligned} \tag{5}$$

The projection of $\Psi(\Phi)$ onto the subspace spanned by the first m eigenvectors is given by :

$$P\Psi(\Phi) = \sum_{k=1}^{m} \beta_k V_k + \bar{\Psi}$$

To obtain an approximate pre-image of $P\Psi(\Phi)$ in the input space, we minimize the error $\rho(\hat{\Phi}) =\| \Psi(\hat{\Phi}) - P\Psi(\Phi) \|^2$. Following the exposition in [15], for a Gaussian kernel (also known as radial basis function) given by :

$$k(\phi_i, \phi_j) = e^{-\frac{d^2(\phi_i, \phi_j)}{2\sigma^2}} \tag{6}$$

where $d^2(\phi_i, \phi_j)$ is a distance measure in the input space, one can obtain an approximate pre-image by setting $\nabla_{\hat{\Phi}} \rho = 0$ and using the approximation $\Psi(\hat{\Phi}) \approx P\Psi(\Phi)$. Here, we directly state the result for finding the pre-image $\hat{\Phi}$ (in the input space χ) of the projection $P\Psi(\Phi)$ [15]:

$$\hat{\Phi} = \frac{\sum_{i=1}^{n} \tilde{\gamma}_i \left(\frac{1}{2}(2 - \tilde{d}^2(P\Psi(\Phi), \Psi(\phi_i)) \right) \phi_i}{\sum_{i=1}^{n} \tilde{\gamma}_i \left(\frac{1}{2}(2 - \tilde{d}^2(P\Psi(\Phi), \Psi(\phi_i)) \right)} \tag{7}$$

where $\gamma_i = \sum_{k=1}^{n} \beta_k a_{ki}$ and $\tilde{\gamma}_i = \gamma_i + \frac{1}{n}(1 - \sum_{j=1}^{n} \gamma_j)$ and $\tilde{d}^2$ can be computed only in terms of the kernel using the following expression [15, 17]:

$$\begin{aligned} \tilde{d}^2(\Psi(\phi_i), P\Psi(\Phi)) = \left(k_\Phi + \frac{1}{n}K\mathbf{1} - 2k_{\phi_i}\right)^T H^T M H \left(k_\Phi - \frac{1}{n}K\mathbf{1}\right) \\ + \frac{1}{n^2}\mathbf{1}^T K \mathbf{1} + K_{ii} - \frac{2}{n}\mathbf{1}^T k_{\phi_i} \end{aligned} \tag{8}$$

where $M = \sum_{k=1}^{n} \frac{1}{\lambda_k} \mathbf{a}_k \, \mathbf{a}_k^T$ and $K_{ii} = k(\phi_i, \phi_i)$.

In this work, we have used the following shape similarity measure given by [18]:

$$d^2(\phi_i, \phi_j) = \int_{p \in Z(\phi_i)} EDT_{\phi_j}(p)dp + \int_{p \in Z(\phi_j)} EDT_{\phi_i}(p)dp, \tag{9}$$

where EDT_{ϕ_i} is the Euclidean distance function of the zero level set of ϕ_i (one can think of it as the absolute value of ϕ_i), and $Z(\phi_i)$ is the zero level set of ϕ_i. This distance measure allows for partial shape matching and was shown [15] to perform better (empirically) than the Euclidean L_2 norm. Note that, $\hat{\Phi}$ is only an approximate pre-image of the projection, since an exact pre-image may not exist.

If we use the kernel $k(\phi_i, \phi_j) =< \phi_i, \phi_j >$, then KPCA is equivalent to doing LPCA. Thus, linear PCA is a particular case of kernel PCA. Choosing the right kernel for a given data set is a topic of active research. In this work we have used the Gaussian kernel (6), which is the most commonly used kernel in the machine learning community.

2.3 Locally linear embedding

The LLE algorithm [19] is based on certain simple geometric principles. Suppose the data consists of n vectors ϕ_i sampled from some smooth underlying manifold. Provided there is sufficient data, we expect each data point and its neighbors to lie on or close to a locally linear patch of the manifold. We can characterize the local geometry of these patches by a set of coefficients that reconstruct each data point from its neighbors. In the simplest formulation of LLE, one identifies k nearest neighbors for a data point. Reconstruction error is then measured by the cost function: $E(W) = \left(\Phi - \sum_j w_j \phi_j\right)^2$. We seek to minimize the reconstruction error $E(W)$, subject to the constraint that the weights w_j that lie outside the neighborhood are zero and $\sum_j w_j = 1$. With these constraints, the weights for points in the neighborhood of Φ can be obtained as [20]:

$$E(W) = \left(\Phi - \sum_{j=1}^{k} w_j \phi_j\right)^2 = \sum_{j=1}^{k}\sum_{m=1}^{k} w_j w_m Q_{jm} \Rightarrow w_j = \frac{\sum_{m=1}^{k} R_{jm}}{\sum_{p=1}^{k}\sum_{q=1}^{k} R_{pq}},$$

$$\text{where } \; Q_{jm} = (\Phi - \phi_j)^T(\Phi - \phi_m) \quad \text{and} \quad R = Q^{-1}. \tag{10}$$

In applications where dimensionality reduction is the major objective, one proceeds further and computes a low dimensional vector corresponding to each ϕ_i, preserving the neighborhood structure by keeping the weights w_j constant [20]. This work uses LLE only for obtaining the neighborhood structure in the training set and not for dimensionality reduction. Thus, we assume that a closed surface S can be represented by a linear combination of its k nearest neighbors. Stacking all the columns of ϕ_i one below the other, one can obtain a vector of dimension D^2, if ϕ_i is of dimension $D \times D$. Thus, given a test point Φ, one can obtain the weights using equation (10) that minimize the reconstruction error $E(W)$. The nearest neighbors are obtained from the training set by finding the squared distance d^2 (equation 9) between Φ and each of the shapes ϕ_i in the training set.

2.4 Kernel LLE

Mercer kernels have been used quite successfully for learning in Support Vector Machines (SVM) and in KPCA as mentioned before. The above LLE algorithm can be generalized for nonlinear manifolds by employing the kernel trick [14]. In [21], the author compares the discriminative power of LLE, KLLE and LPCA by projecting the training data to a lower dimensional space and thereby comparing the recognition rate of a given test sample. The methods presented in this work are quite different than those proposed in [21], since we do not compute a low dimensional data for LLE or KLLE, but compare their performances in the input space itself. This is quite essential for shape analysis in which one needs to compute how accurately a given data point can be reproduced in the input space using these techniques. Thus, the method proposed in [21] uses LLE, KLLE only for classification purposes, while we utilize it to see its performance in the input space. A major contribution of this work is the formulation of a method to find the pre-image of the projection in the kernel space, given the fact that we do not know the mapping Ψ.

The basic idea behind KLLE is to minimize the error (given a test point Φ) $E(W) = \left(\Psi(\Phi) - \sum_j w_j \Psi(\phi_j)\right)^2$. Proceeding as shown in LLE before, we get the following expression for the weights:

$$w_j = \frac{\sum_{m=1}^{k} R_{jm}}{\sum_{p=1}^{k}\sum_{q=1}^{k} R_{pq}} \quad \text{where,}$$

$$\begin{aligned} Q_{jm} &= (\Psi(\Phi) - \Psi(\phi_j))^T(\Psi(\Phi) - \Psi(\phi_m)) = k(\Phi,\Phi) - k(\Phi,\phi_m) \\ &\quad - k(\Phi,\phi_j) + k(\phi_j,\phi_m) \quad \text{and} \quad R = Q^{-1}. \end{aligned} \tag{11}$$

The weights w_j so obtained minimize the error $E(W)$ in the feature space F, i.e., $\Psi(\Phi) = \sum_{j=1}^{k} w_j \Psi(\phi_j) + \sqrt{E} = \Psi(\hat{\Phi}) + \sqrt{E}$. Assuming E to be small, we have $\Psi(\Phi) \approx \Psi(\hat{\Phi})$. Our goal now is to find the pre-image of $\Psi(\Phi)$. However, an exact pre-image of $\Psi(\Phi)$ may not exist [16], hence we find an approximate pre-image of $\Psi(\Phi)$ in the input space χ. Thus, we want to find the point $\Psi(z)$

which is closest to $\Psi(\Phi)$ and for which the pre-image can be computed. This can be achieved by minimizing the following:

$$\rho(z) = \| \Psi(z) - \Psi(\Phi) \|^2 \approx k(z,z) - 2\sum_j w_j k(z,\phi_j) + k(\Phi,\Phi),$$

where we have substituted the approximation for $\Psi(\Phi)$. Setting $\nabla_z \rho(z) = 0$ and using the kernel $k(z,\Phi) = \exp(-\frac{\|z-\Phi\|^2}{2\sigma^2})$, one gets the following expression for finding z:

$$z = \frac{\sum_{j=1}^{k} w_j k(z,\phi_j)\ \phi_j}{\sum_{j=1}^{k} w_j k(z,\phi_j)} \tag{12}$$

This equation contains z on both sides of the equation and hence can be solved by fixed-point iteration technique. However, the solution will depend on the starting point and will be very susceptible to local minima. A unique (but approximate) solution to z can be found by noting that

$$k(z,\phi_j) = <\Psi(z), \Psi(\phi_j)> \approx <\Psi(\Phi), \Psi(\phi_j)> = \ k(\Phi,\phi_j)$$

where we assume $\Psi(\Phi) \approx \Psi(z)$. Note that this assumption is valid since we are trying to find the point $\Psi(z)$ that is closest to $\Psi(\Phi)$. The error in the computed pre-image will be proportional to the error in approximating $\Psi(z) = \Psi(\Phi)$, which in general can be assumed to be small. As shown in [15], better results can be obtained if the distance measure (9) (for d^2) is used in the Gaussian kernel instead of the Euclidean L_2 norm and hence we use it in all our experiments as described in the next section.

A pre-image can be computed not only for a Gaussian kernel, but for any invertible kernel. If we assume a polynomial kernel $k(\phi_i,\phi_j) = \left(c + \phi_i^T \phi_j\right)^d$, where d is the degree of the polynomial and c is any constant, then the pre-image z of a point $\Psi(\Phi)$ is given by

$$z = \frac{\sum_{j=1}^{k} w_j k(\Phi,\phi_j)^{\frac{d-1}{d}}\ \phi_j}{k(\Phi,\Phi)^{\frac{d-1}{d}}} \tag{13}$$

Thus, LLE is a particular case of KLLE with a polynomial kernel of degree $d = 1$ and $c = 0$. Once again, the k nearest neighbors can be computed using the distance relation (9) or any other metric on the space of shapes [22, 11, 7, 23–25].

3 Experiments

In this section, we describe two experiments to test how well each method performs given a training set of shapes. The first set of 3D shapes consists of the left caudate nucleus and the second set consists of the left hippocampus. These are structures in the brain for which a shape prior is often used in segmentation algorithms. A typical measure to test the performance of these methods is to

see how well an unknown shape gets projected by each of these methods. In this work, a quantitative measure was calculated by finding the number of voxels that got mislabelled, i.e., by finding the set symmetric difference between the projection and the original test shape.

The training set for the caudate nuclei consisted of 26 elements, each of them embedded in a signed distance function. Figure 1 shows a few shapes in the training set. In Figure 2, an "unseen shape" (i.e., a shape not in the training set) is shown and also the pre-image of the projection using each of the methods. Table 1 shows the number of mislabeled voxels for each of the methods. For LPCA and KPCA, 20 coefficients were used in finding the projection while for LLE and KLLE 20 nearest neighbors were used so that we do not obtain biased results in favor of a particular method. Clearly, the kernel methods perform better than their linear counterparts. More specifically, KLLE performs almost as well or better than KPCA, but with a smaller computational burden.

Table 1. Mislabelled voxels for left caudate nucleus

Volume	Volume Size	LPCA	LLE	KPCA	KLLE
1	2750	119	50	**37**	42
2	3774	134	105	92	**81**
3	2489	108	66	57	**52**

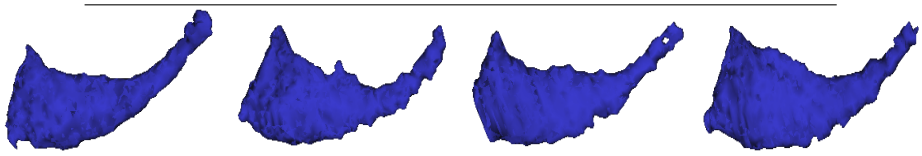
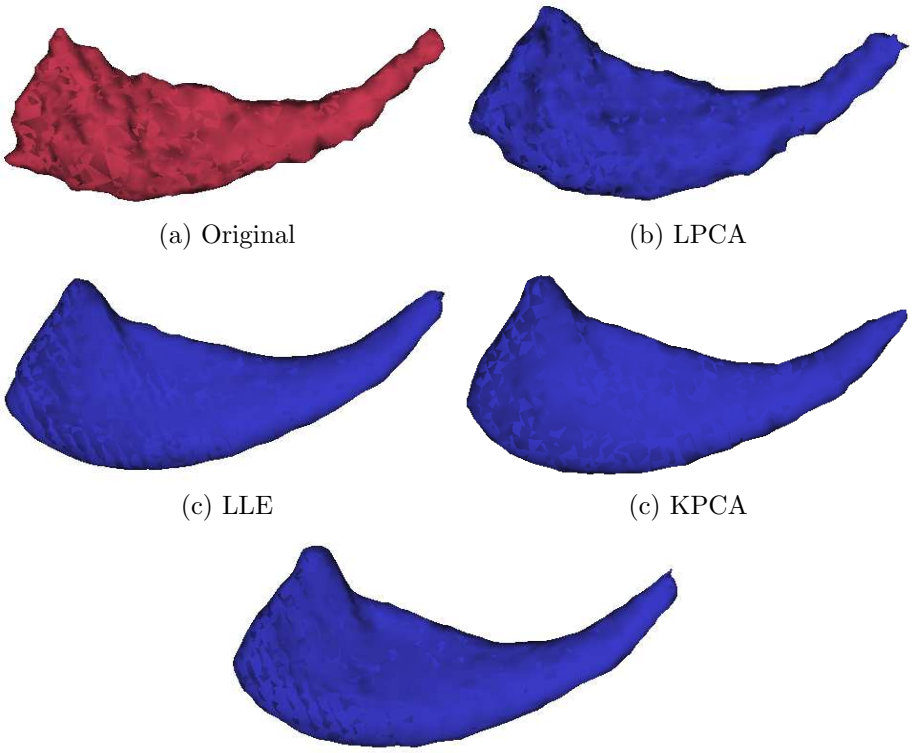


Fig. 1: Sample shapes of left Caudate nucleus from the training set.

The second training set of the hippocampii data contained 22 elements. Figure 3 shows a few shapes from the training set and figure 4 shows the original and pre-images of projection for each of the methods. For this experiment, we used 15 coefficients for LPCA and KPCA and 15 nearest neighbors for LLE and KLLE. Table 2 gives the number of mislabelled voxels for each of the methods. Figure 5 shows the weights assigned to each of the neighbors (for all the three test shapes) using LLE and KLLE. Clearly, KLLE assigns larger weights to points closer to the test shape than to points farther away. Thus, only points in the locally linear patch of the feature space are assigned significant weights, whereas other points are assigned weights close to zero. This nonlinear distribution is expected since we used a Gaussian kernel. Once again, it is clear that KLLE performs better than all the other methods. It should be noted that, LLE and KLLE can perform even better with the proper choice of the number of nearest neighbors as given in [20]. To make a fair assessment of each method, we kept k (nearest neighbors) fixed and did not optimize the algorithm as given in [20].

Table 2. Mislabelled voxels for left hippocampus

Volume	Volume Size	LPCA	LLE	KPCA	KLLE
1	1117	440	378	322	**296**
2	1108	306	258	212	**205**
3	1568	804	574	494	**371**



(a) Original (b) LPCA

(c) LLE (c) KPCA

(e) KLLE

Fig. 2: Projection of left Caudate nucleus (Volume 1) using each of the methods.

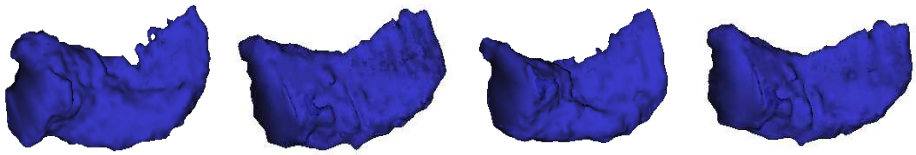
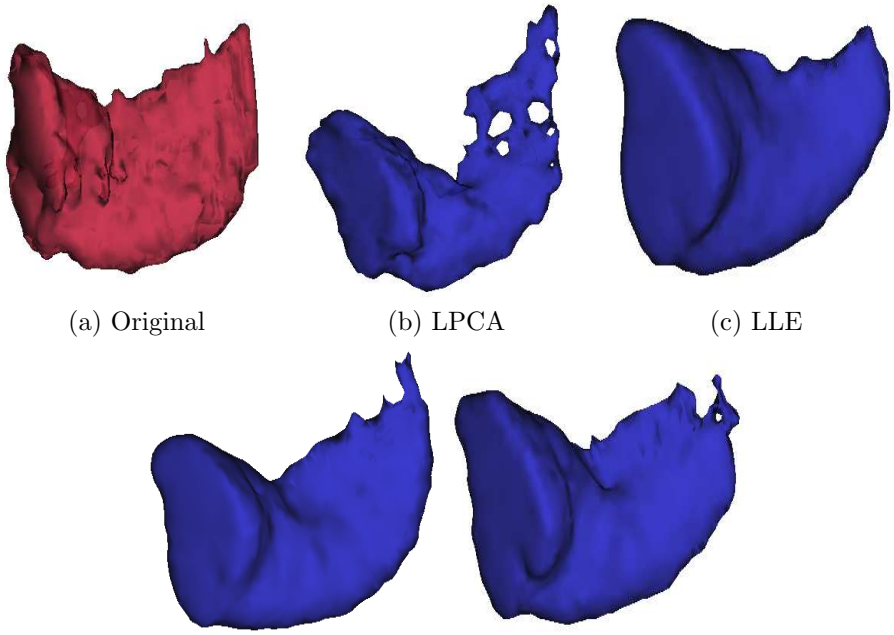


Fig. 3: Sample shapes of left hippocampus from the training set.

In all of the experiments above, the parameter σ used in the Gaussian kernel was fixed to be some function of the average minimum distance between shapes in the training set [8], i.e., $\sigma^2 = c\ \frac{1}{n}\sum_{i=1}^{n} \min_{j\neq i} d^2(\phi_i, \phi_j)$., where c is a user defined real number. The training data (hand segmented shapes) was obtained from the NAMIC data repository of the Brigham and Women's Hospital, Boston, MA. The entire code was written in C++ using the ITK and VTK libraries.

4 Remarks

In this paper, we have proposed a new algorithm for finding an approximate pre-image of a point in the kernel space in the context of Kernel LLE which is a generalization of LLE to the kernel space. We have compared this method



(a) Original (b) LPCA (c) LLE

(d) KPCA (e) KLLE

Fig. 4: Projection of left Hippocampus (Volume 3) using each of the methods.

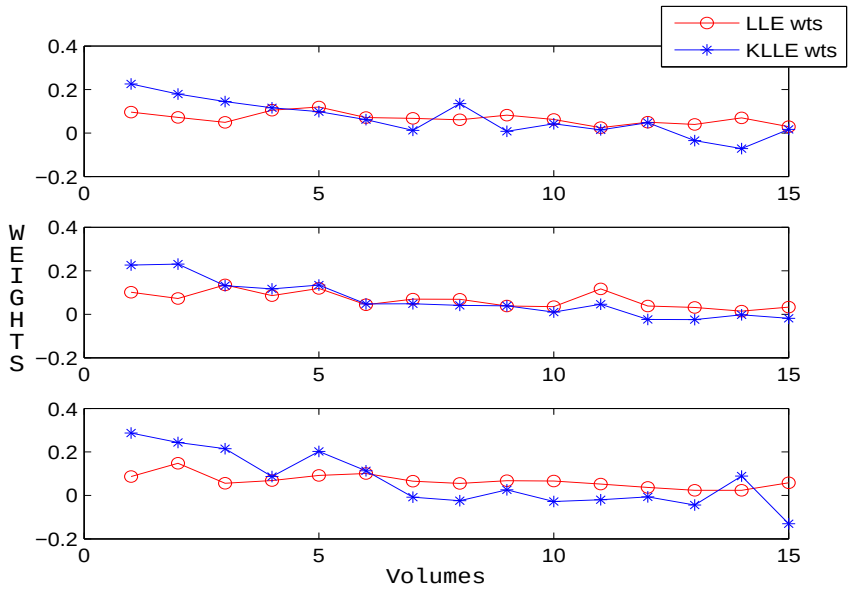


Fig. 5: Weights assigned to the 15 nearest neighbors by LLE and KLLE for each of the test shapes of hippocampus. On the x-axis, 1 is the closest neighbor, 15 is the farthest.

with other methods such as linear PCA, kernel PCA and LLE in terms of its capability to represent unseen shapes. Experiments show that it performs better than LPCA and LLE and is comparable to KPCA, but with considerably fewer computations. We certainly do not claim that KLLE is the best method to use for any given training set of shapes, but it did give good results on the training data on which it was tested.

Nevertheless, representing a shape using its nearest neighbors requires that the training set contain sufficient data points. LPCA and KPCA have an innate capability to "produce" shapes by varying the PCA coefficients. This is not the case with LLE or KLLE. On the other hand, if sufficient amount of data is available, LLE and KLLE can perform better than PCA based algorithms. Another advantage of LLE and KLLE is that they allow one to learn shapes of completely different geometries, within the same training set. The reason for this is that these methods use only their nearest neighbors to find the projection instead of using the entire training set which is the case with KPCA and LPCA. One of the reasons why the kernel methods work better than their linear counterparts is that, the set of signed distance functions (SDF) is not closed under addition. Thus, the variations captured by linear methods are the variations in the SDF's and not in the embedded shapes, whereas the kernel methods capture the variations in shapes and not the embeddings. We should also note that, the performance of all these methods will get better if one has a large training set (with shapes of the same object).

In this work, we have used signed distance function to represent shapes. However, the algorithms used here do not depend on any particular type of representation. Performing a detailed comparative analysis using all of these methods with different representations (parametric and implicit) for shapes is the subject of future research. We would also like to test these methods on a wide variety of shapes with varying sizes of the training data set.

References

1. Terzopoulos, D., Szeliski, R.: Tracking with Kalman Snakes. In: Active Vision. MIT Press (1992) 3–20
2. Ayache, N., Cohen, I., Herlin, I.: Medical image tracking. In: Active Vision. MIT Press (1992) 3–20
3. Blake, A., Yuille, A., eds.: Active Vision. MIT Press (1992)
4. Osher, S., Paragios, N.: Geometric Level Set Methods in Imaging, Vision and Graphics. Springer Verlag (2003)
5. Dambreville, S., Rathi, Y., Tannenbaum, A.: Shape-based approach to robust image segmentation using kernel pca. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2006)
6. Cootes, T., Taylor, C., Cooper, D., Graham, J.: Active shape models, their training and application. In: Computer Vision and Image Understanding. Volume 61. (1995) 38–59
7. Faugeras, O., Gomes, J.: Dynamic shapes of arbitrary dimension: the vector distance functions. In Cipolla, R., Martin, R., eds.: Proceedings of the Ninth IMA

Conference on Mathematics of Surfaces. The Mathematics of Surfaces IX, Springer (2000)
8. Cremers, D., Kohlberger, T., Schnorr, C.: Nonlinear shape statistics in mumford-shah based segmentation. In: 7th ECCV '02. Volume 2351. (2002) 93–108
9. Leventon, M., Grimson, E., Faugeras, O.: Statistical shape influence in geodesic active contours. In: Proc. CVPR, IEEE (2000) 1316–1324
10. Tsai, A., Yezzi, A., Wells, W., Tempany, C., Tucker, D., Fan, A., Grimson, E., Willsky, A.: A shape-based approach to the sementation of medical imagery using level sets. IEEE Trans. on Medical Imaging **22** (2003) 137–153
11. Gomes, J., Faugeras, O.: Shape representation as the intersection of $n - k$ hypersurfaces. Technical Report 4011, INRIA (2000)
12. Osher, S.J., Sethian, J.A.: Fronts propagation with curvature dependent speed: Algorithms based on hamilton-jacobi formulations. Journal of Computational Physics **79** (1988) 12–49
13. Wang, Y., Staib, L.: Boundary finding with correspondence using statistical shape models. In: Proc. CVPR. (1998) 338–345
14. Scholkopf, B., Smola, A., Muller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Technical report, Max-Planck-Institute fur biologische Kybernetik (1996)
15. Rathi, Y., Dambreville, S., Tannenbaum, A.: Statistical shape analysis using kernel pca. In: SPIE, Electronic Imaging. (2006)
16. Mika, S., Scholkopf, B., Smola, A., K.R.Muller, M.Scholz, G.Ratsch: Kernel pca and de-noising in feature spaces. (Advances in Neural Information Processing Systems 11)
17. Kwok, J., Tsang, I.: The pre-image problem in kernel methods. In: 20th Intl. Conference on Machine Learning. (2003)
18. Funkhouser, T., Kazhdan, M., Shilane, P., Min, P., Kiefer, W., Tal, A., Rusinkiewicz, S., Dobkin, D.: Modeling by example. In: ACM Transactions on Graphics (SIGGRAPH 2004). (2004)
19. Saul, L.K., Roweis, S.: An introduction to locally linear embedding. (http://www.cs.toronto.edu/ roweis/lle/papers/lleintro.pdf)
20. D.Ridder, Duin, R.: Locally linear embedding for classification. Technical Report PH-2002-01, Pattern Recognition Group, Delft University of Technology (2002)
21. DeCoste, D.: Visualizing mercer kernel feature spaces via kernelized locally-linear embeddings. In: 8th Intl. Conf. on Neural Information Processing. (2001)
22. Chan, T., Zhu, W.: Level set based shape prior segmentation. Technical report, Computational Applied Mathematics, UCLA (2003)
23. Srivastava, A., Joshi, S., Mio, W., Liu, X.: Statistical shape analysis: Clustering, learning and testing. Trans. PAMI (2005)
24. Yezzi, A., Mennucci, A.: Metrics in the space of curves. http://arxiv.org/abs/math.DG/0412454 (2004)
25. Michor, P., Mumford, D.: Riemannian geomtries of space of plane curves. (http://front.math.ucdavis.edu/math.DG/0312384.)